IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 20, Issue 6, Ver. II (Nov - Dec 2018), PP 60-68
www.iosrjournals.org

Cryptanalysis and Improvement of Kang et al. Certificateless Aggregate Signature Scheme

N. B. Gayathri[1], P. Vasudeva Reddy[2]
[1](*Department of Engineering Mathematics, Andhra University, India)*
[2](*Department of Engineering Mathematics, Andhra University, India)*
Corresponding Author: N. B. Gayathri

Abstract: *The most important contribution of modern cryptography is the invention of digital signatures. To deal with specific application scenarios, digital signature schemes have been evolved with different variants. One of such variant is aggregate signature scheme, which allows aggregation of different signatures by different users on different messages, to achieve computational and communication efficiency. Such schemes are useful in the design of Wireless Sensor Networks (WSN), Mobile Ad-hoc Networks (MANETS), and Vehicular Ad-hoc Networks (VANETS); where storage, bandwidth and computational complexity are major constraints. Recently, in 2017, B. Kang et al. proposed an efficient certificateless aggregate signature scheme in which they claimed that their scheme is secure against Type I and Type II adversary. However, we find some vulnerability in the signature generation algorithm. In this paper we show that this scheme is insecure against type II adversary i.e. a malicious Key Generation Centre (KGC) can forge a legal aggregate signature on any message without any access to user's secret information. Further, we proposed an improved Certificateless Aggregate Signature (CLAS) scheme. The proposed CLAS scheme is using bilinear pairings over elliptic curves and is proven secure in Random Oracle Model under the assumption of Computational Diffie-Hellman Problem is hard. The concrete security proof assures that our scheme is secure against Type I and Type II adversary. We compared our scheme with well known existing schemes. Efficiency analysis shows that our scheme is more efficient than existing schemes in terms of communication and computational costs.*

Date of Submission: 26-11-2018 Date of acceptance: 07-12-2018

I. Introduction

Digital signature is the cornerstone to secure all electronic transactions in our digital world. Digital signature scheme are used widely for providing data integrity, authentication and non-repudiation for digital communications. The security assurance of the traditional Public Key Cryptography (PKC) [1] is based on the certificate, signed by a Certification Authority (CA), containing the relationship between the key pairs, i.e., a public key and a private key, and the user's identity. But certificate management leads to extra storage, large computation and communication costs. Contrast with traditional PKC, Identity Based public key cryptography (ID-PKC) [2] does not need any certificate to ensure the authenticity of public/private key pair. In this system, public key of a user is derived from the user's identity and secret key is generated by a trusted third party called Private Key Generator (PKG). Though ID-PKC successively removed the necessity of certificates, it suffers from inherent key escrow problem. To overcome afore mentioned difficulties in PKC and ID-PKC, Al-Riyami [3] presented a new structure called Certificateless Public Key Cryptography (CL-PKC) in 2003. In this system, the full private key of a user is divided into two parts. The first part, called partial private key, is controlled by a Key Generator Center (KGC). The second part is chosen by the user himself and remains secret to the KGC. Hence this cryptosystem not only enjoys the advantages of ID-based cryptography but also eliminates the key escrow problem by setting user's full secret key as a combination of partial private key generated by KGC and some secret value chosen by the user.

Based on the potential adversaries' behaviour, we consider the following two types of attacks. We discuss the security issues of CL-PKC by depending on which part of the private key is compromised.

1. Type I: Key Replacement Attack: The Adversary cannot access s but capable to replace the public key of any user with a value of his choice.
2. Type II: Malicious KGC Attack: The Adversary can access s but cannot change the public key of any user.

To deal with different scenarios, digital signature schemes have evolved into many different variants. One of such variants is aggregate signature. The aggregate signature scheme allows n signatures on n distinct messages from n distinct users to aggregate a single signature. An aggregate signature scheme enables us to achieve high efficiency by reducing either communicational overhead or computational cost or both. The

concept of aggregate signature was first introduced by Boneh et al. [4].The aggregate signature schemes could be used in many applications such as wireless sensor networks, secure routing protocols, banking transactions, traffic control, military applications etc., where storage, bandwidth and computational complexity are of major constraints. There are many CLAS schemes were proposed in the literature. Recently B. Kang et al. [5] proposed an efficient CLAS scheme and claimed that their scheme is secure against Type I and Type II adversary. However we found that their proposed scheme is insecure against Type II attack. Here we discuss some important drawbacks of their scheme which led us to write this paper.

1. In the Setup algorithm they used three map to point hash functions which enormously increase the computational cost.
2. In Signature Generation algorithm, the second part of the signature i.e. T_i, was generated without involving the message while using either secret key (x_i) or random key (r_i). This can make an attacker to forge a valid signature on any message.

Due to the above drawbacks, B. Kang et al. scheme [5] is computationally inefficient and vulnerable to Type-II attacks. In this paper, we present the cryptanalysis of B. Kang et al. scheme by demonstrating how a Type II adversary can forge a legal aggregate signature on any message without any access to user's secret information. To overcome these drawbacks and to resist against the mentioned attacks, we improve the B. Kang et al. scheme and present a concrete security proof. Our improved scheme is more efficient and secure against the above said attacks.

Related Work: The first Certificateless Aggregate Signature (CLAS) scheme was proposed by Castro et al. [6] in 2007. After that, many secure CLAS schemes have been proposed. In 2009, Zhang et al. [7] proposed a new CLAS scheme. In 2010, Gong et al. [8] proposed a practical CLAS from bilinear maps. In 2012, Chen et al. [9] presented an efficient CLAS scheme. In 2014, Zho et al. [10] presented a compact CLAS scheme. In the same year Zhang et al. [11] presented a security analysis of CLAS scheme. In 2016, Kang et al. [12] proposed a secure CLAS scheme. In the same year Kar et al. [13] proposed a short CLAS scheme. But all these schemes require relatively more number of pairing operations in verification process and these operations increases linearly with the number of signers in aggregation (verification) process and deviates from the goal of aggregate signatures. Later, many researchers came up with CLAS scheme with fixed paring operations in aggregate verification. In 2010, Zhang et al. [14] proposed an efficient CLAS scheme with constant pairings. In 2013, Xiong et al. [15] proposed an efficient CLAS scheme with constant pairings. But Cheng et al. [16] presented a cryptanalysis on Xiong et al. [15] scheme and proposed an improved scheme. In 2014, Chen et al. [17] and Liu et al. [18] individually proposed different CLAS schemes with efficient verification. In the same year, Tu et al. [19] presented a reattack on Xiong et al. [15] scheme. In 2015, Deng et al. [20] and Chen et al. [21] individually proposed two different secure CLAS schemes. In 2015, Horng et al. [22] proposed an efficient CLAS scheme for vehicular sensor networks. In the same year, Malhi et al. [23] presented an efficient CLAS scheme for vehicular Ad-hoc networks. But most of the schemes [15,17,18,22] are insecure against different types of attacks. Recently B. Kang et. al. [5] proposed an efficient certificateless aggregate signature scheme and claimed that their scheme is secure against Type I and Type II adversary.

Our Contributions: In this paper, we present the cryptanalysis of B. Kang et al. scheme [5] by demonstrating how a Type II adversary can forge a legal aggregate signature on any message without knowing any user's secret information. Later, we improve B. Kang et al. [5] signature scheme and prove that the improved scheme is efficient and secure against Type I and Type II adversary.

Organization: The remaining part of this paper is structured as follows. Section II presents a brief review of the Kang et al.'s scheme. The cryptanalysis of Kang et al scheme is presented in Section III. In Section IV we present our improved CLAS scheme with security analysis. Section V presents efficiency analysis. Finally Section VI deals with conclusion.

II. Review of Kang et al. Certificateless Aggregate Signature

In order to facilitate cryptanalysis, we follow the notations from [5]. CLAS scheme for B. Kang et al. [5] includes the following algorithms.

Setup: Given a security parameter τ, the KGC selects an additive group G_1 and a multiplicative cyclic group G_2 with the same order q, and chooses a bilinear map $e: G_1 \times G_1 \to G_2$. Then, KGC selects a generator of G_1, P and random $s \in Z_q^*$ as the master key and sets system public key $P_{Pub} = sP$. KGC also picks four cryptographic Hash functions. $H_1, H_3, H_4 : \{0,1\}^* \to G_1$, $H_2 : \{0,1\}^* \times \{0,1\}^* \times G_1 \times G_1 \to Z_q^*$. The system parameter list is

$params = (G_1, G_2, e, P, P_{Pub}, H_1, H_2, H_3, H_4)$.

Partial-Private-Key-Extract: KGC generates the partial private key D_i for the user with identity ID_i as follows:

1. Calculate $Q_i = H_1(ID_i)$.
2. Output $D_i = sQ_i$.

User Key Gen: This algorithm selects a random $x_i \in Z_q^*$ as one user's secret value, and generates the user's public key as $P_i = x_i P$.

Signature Generation: Given state information w, one user U_i with identity ID_i and public key P_i signs a message m_i as follows.

1. Select a random number $r_i \in Z_q^*$, calculates $R_i = r_i P$.
2. Calculate $h_i = H_2(m_i, ID_i, P_i, R_i), Z = H_3(w), F = H_4(w)$, and $T_i = h_i D_i + x_i Z + r_i F$.
3. Output signature $\sigma_i = (R_i, T_i)$.

Aggregate: For n message-signature pairs $(m_1, \sigma_1 = (R_1, T_1))$ $(m_2, \sigma_2 = (R_2, T_2)), ..., (m_n, \sigma_n = (R_n, T_n))$ from n users $U_1, U_2, ..., U_n$ (who has the same state information), respectively, any aggregate signature generator can compute $T = \sum_{i=1}^{n} T_i$ and output the aggregate signature $\sigma = (R_1, R_2, ..., R_n, T)$.

Aggregate Verify: To verify an aggregate signature $\sigma = (R_1, R_2, ..., R_n, T)$ on messages $m_1, m_2, ...m_n$ from n users $U_1, U_2, ..., U_n$ with identities $ID_1, ID_2, ...ID_n$, corresponding public keys $P_1, P_2, ..., P_n$, and same state information w, the verifier does the following steps:

1. Calculate $Q_i = H_1(ID_i)$, $h_i = H_2(m_i, ID_i, P_i, R_i)$, for all $i, 1 \le i \le n$, $Z = H_3(w), F = H_4(w)$.
2. Verify $e(T, P) = e(P_{Pub}, \sum_{i=1}^{n} h_i Q_i) e(Z, \sum_{i=1}^{n} P_i) e(F, \sum_{i=1}^{n} R_i)$.
3. If the equation holds, output true. Otherwise, output false.

III. Cryptanalysis for Kang et al. Certificateless Aggregate Signature Scheme

In this section we show the scheme proposed by B. Kang et al. [5] is forgeable under Type II attack.

Let ADV_2 be a malicious KGC, who knows the Partial secret key D_i of a user with identity ID_i. Now adversary ADV_2 can issue a sign query to obtain the signature σ_i on a message m_i and a state of string w such that $\sigma_i = (R_i, T_i)$ where $R_i = r_i P$ and $T_i = h_i D_i + x_i Z + r_i F$ and $h_i = H_2(m_i, ID_i, P_i, R_i)$. Note that the ADV_2 does not know the user's secret value x_i directly. However, from σ_i, ADV_2 can compute $T_i - h_i D_i = x_i Z + r_i F = S_i$ (say) since D_i is known.

Now ADV_2 can forge a signature $\sigma_i' = (R_i', T_i')$ for any message m_i' under the same state string $'w'$. For that, ADV_2 sets $R_i = R_i'$ and $T_i' = S_i + h_i' D_i$ where $h_i' = H_2(m_i', ID_i, P_i, R_i')$.

Since

$e(T_i', P)$

$= e(S_i + h_i' D_i, P)$

$= e(x_i Z + r_i F + h_i' D_i, P)$

$= e(P_{Pub}, h_i' Q_i) e(Z, P_i) e(F, R_i')$.

Hence $\sigma_i' = (R_i', T_i')$ is indeed a valid signature for the message m_i'.

Similarly, ADV_2 can forge all the users' signatures on messages m_i' for $i = 1, 2, 3, ...n$ and aggregates. Then the forged aggregate signature is a valid aggregate signature for the message m_i'.

Discussion: The above attack on Kang et al. [5] scheme is possible because of the following reason. In our attack, malicious KGC is able to compute $x_i Z + r_i F$ without knowing the values of x_i and r_i. i.e. KGC can compute one segment of the second part of signature which is associated with user's secret value and random key. This is possible, since these keys are not associated with the signing message. Hence KGC can forge the

signature $\sigma_i = (R_i, T_i)$ on any message, since the term $x_i Z + r_i F$ is not associated with the corresponding message.

IV. Improvement of Kang et al. CLAS Scheme

To resist the above attack, the Kang et al. scheme can be modified as follows.

Improved CLAS Scheme:

1. **Setup:** Given a security parameter τ, the KGC selects an additive group G_1 and a multiplicative cyclic group G_2 with the same order q, and chooses a bilinear map $e: G_1 \times G_1 \to G_2$. Then, KGC selects a generator of G_1, P and random $s \in Z_q^*$ as the master key and sets system public key $P_{Pub} = sP$. KGC also picks four cryptographic Hash functions. $H_1, H_4 : \{0,1\}^* \to G_1$, $H_2, H_3 : \{0,1\}^* \to Z_q^*$.
   The system parameter list is params = $(G_1, G_2, e, P, P_{Pub}, H_1, H_2, H_3, H_4)$.
2. **Partial-Private-Key-Extract:** KGC generates the partial private key D_i for the user with identity ID_i as follows:
   i. Calculate $Q_i = H_1(ID_i)$.
   ii. Output $D_i = sQ_i$.
3. **User Key Gen:** This algorithm selects a random $x_i \in Z_q^*$ as one user's secret value, and generates the user's public key as $P_i = x_i P$.
4. **Signature Generation:** Given state information w, one user U_i with identity ID_i and public key P_i signs a message m_i as follows.
   i. Select a random number $r_i \in Z_q^*$, calculates $R_i = r_i P$.
   ii. Calculate $h_i = H_2(m_i, ID_i, P_i, R_i)$, $g_i = H_3(m_i, ID_i, P_i, R_i), F = H_4(w)$.
   iii. Calculate $T_i = h_i D_i + (g_i x_i + r_i)F$.
   iv. Output signature $\sigma_i = (R_i, T_i)$.
   Note that users never collude with each other.
5. **Aggregate:** For n message-signature pairs $(m_1, \sigma_1 = (R_1, T_1))$ $(m_2, \sigma_2 = (R_2, T_2)),...,$ $(m_n, \sigma_n = (R_n, T_n))$ from n users $U_1, U_2, ..., U_n$ (who has the same state information), respectively, any aggregate signature generator can compute $T = \sum_{i=1}^{n} T_i$ and output the aggregate signature $\sigma = (R_1, R_2, ..., R_n, T)$.
6. **Aggregate Verify:** To verify an aggregate signature $\sigma = (R_1, R_2, ..., R_n, T)$ on messages $m_1, m_2, ...m_n$ from n users $U_1, U_2, ..., U_n$ with identities $ID_1, ID_2, ...ID_n$, corresponding public keys $P_1, P_2, ..., P_n$, and same state information w, the verifier does the following steps:
   i. Calculate $Q_i = H_1(ID_i)$, $h_i = H_2(m_i, ID_i, P_i, R_i)$, $g_i = H_3(m_i, ID_i, P_i, R_i)$ for all $i, 1 \le i \le n$, and $F = H_4(w)$.
   ii. Verify $e(T, P) = e(P_{Pub}, \sum_{i=1}^{n} h_i Q_i) e(F, \sum_{i=1}^{n} R_i + g_i P_i)$.
   iii. If the equation holds, output true. Otherwise, output false.

Security Analysis: The improved CLAS scheme is secure against Type I and Type II adversaries. The security proof is similar to that of Kang et al. scheme.

Theorem 1: The proposed certificateless aggregate scheme is existentially unforgeable against Type I adversary under the assumption that the CDH problem is intractable.

Proof: To prove the proposed scheme is existentially unforgeable against Type I adversary, we show how a CDH attacker N_1 uses Type I adversary A_1 to compute abP from (P, aP, bP).

Setup: N_1 sets system public $P_{Pub} = ap$ and params $= (G_1, G_2, e, P, P_{Pub}, H_1, H_2, H_3, H_4)$ and sends params to A_1. A_1 execute the following types of queries in an adaptive manner.

H_1 queries: There is a list H_1^{list} to record H_1 queries. When A_1 queries $H_1(ID_i)$, the same answer will be given if the query can be found on H_1^{list}. Otherwise, N_1 sets $Q_i = \varepsilon_i \in Z_q^*$ at random and flips a coin $c_i \in \{0,1\}$. If $c_i = 0$, N_1 sets $Q_i = \varepsilon_i(bP)$, adds $(ID_i, \perp, Q_i, c_i)$ to H_1^{list} and returns Q_i. Otherwise, N_1 sets $Q_i = \varepsilon_i P$, adds $(ID_i, \varepsilon_i, Q_i, c_i)$ to H_1^{list} and returns Q_i.

H_2 queries: There is a list H_2^{list} to record H_2 queries. When A_1 queries $H_2(m_i, ID_i, P_i, R_i)$, the same answer will be given if the query can be found on H_2^{list}. If the query cannot be found on H_2^{list}, N_1 picks a random $\delta_i \in Z_q^*$, adds $(m_i, ID_i, P_i, R_i, \delta_i)$ to H_2^{list} and then returns δ_i.

H_3 queries: There is a list H_3^{list} to record H_3 queries. When A_1 queries $H_3(m_i, ID_i, P_i, R_i)$, the same answer will be given if the query can be found on H_3^{list}. If the query cannot be found on H_3^{list}, N_1 picks a random $\phi_i \in Z_q^*$, adds $(m_i, ID_i, P_i, R_i, \phi_i)$ to H_3^{list} and then returns ϕ_i.

H_4 queries: There is a list H_4^{list} of tuples (w_i, μ_i, F_i) to record H_4 queries. When A_1 queries $H_1(w_i)$, the same answer will be given if the query can be found on H_4^{list}. Otherwise N_1 picks a random $\mu_i \in Z_q^*$, calculates $F_i = \mu_i P$, adds (w_i, μ_i, F_i) to H_4^{list} and then returns F_i.

Partial-Private-Key queries: N_1 keeps a list K^{list} to Partial-Private-Key queries. When A_1 queries a Partial-Private-Key and the query can be found on K^{list}, N_1 first does a H_1 query on ID_i and finds the tuple $(ID_i, \varepsilon_i, Q_i, c_i)$ on then N_1 does as follows.

1. If $c_i = 0$, N_1 aborts.
2. Else if there is a tuple (ID_i, x_i, D_i, P_i) on K^{list}, N_1 sets and returns D_i.
3. Otherwise, calculates $D_i = \varepsilon_i P_{Pub}$, set $x_i = P_i = \perp$, return D_i as answer and add (ID_i, x_i, D_i, P_i) on K^{list}.

Public-Key queries: To answer a Public-Key query on ID_i, if the query can be found on K^{list}, the same answer will be given. Otherwise N_1 does as follows.

1. If there is a tuple (ID_i, x_i, D_i, P_i) on K^{list} (in this case, the public key P_i of ID_i is $\perp$), choose $x_i' \in Z_q^*$, compute $P_i' = x_i' P$, return P_i' as answer and update (ID_i, x_i, D_i, P_i) to (ID_i, x_i', D_i, P_i').
2. Otherwise, select $x_i \in Z_q^*$ at random, calculate $P_i = x_i P$, return P_i as answer, set $D_i = \perp$ and add (ID_i, x_i, D_i, P_i) to K^{list}.

Secret-Value queries: On receiving a Secret-Value query on ID_i, firstly, N_1 makes Public-Key query on ID_i, then finds (ID_i, x_i, D_i, P_i) on list K^{list}, returns x_i. (Note that the value of x_i may be $\perp$).

Public-Key-Replacement queries: When N_1 receives a Public-Key-Replacement query, N_1 first finds (ID_i, x_i, D_i, P_i) on list K^{list}, if such tuple does not exist on K^{list} or $P_i = \perp$, N_1 first makes Public-Key query on ID_i, then updates P_i to P_i'.

Sign queries: When N_1 receives a Sign query on m_i by one user with identity ID_i, firstly N_1 recovers $(ID_i, \varepsilon_i, Q_i, c_i)$ from H_1^{list}, $(m_i, ID_i, P_i, R_i, \delta_i)$ from H_2^{list}, $(m_i, ID_i, P_i, R_i, \phi_i)$ from H_3^{list}, then does as follows.

1. If $c_i = 0$, select $r_i, \mu_i \in Z_q^*$, set $R_i = -\mu_i^{-1}(\delta_i Q_i + P) - \phi_i P_i, F_i = \mu_i P_{Pub}$ and record $(w_i, \perp, \mu_i P_{Pub})$ on list H_4^{list}, then sets $T_i = P_{Pub}$, output $\sigma_i = (R_i, T_i)$. Here δ_i, ϕ_i are taken from H_2^{list}, H_3^{list} respectively.
2. If $c_i = 1$, select $r_i \in Z_q^*$, sets $R_i = r_i P + \phi_i P_i, T_i = \delta_i \varepsilon_i P_{Pub} + \mu_i r_i P$ and output $\sigma_i = (R_i, T_i)$. Here δ_i, ϕ_i, μ_i are taken from H_2^{list}, H_3^{list}, H_4^{list} respectively.

Forgery: A_1 outputs a forged aggregate signature $\sigma^* = \left\{R_1^*, R_2^*, ..., R_n^*, T^*\right\}$ under a set U of n users with identities set $L_{ID}^* = \left\{ID_1^*, ..., ID_n^*\right\}$ and the corresponding public keys set $L_{PK}^* = \left\{P_1^*, ..., P_n^*\right\}$, messages set $L_m^* = \left\{m_1^*, ..., m_n^*\right\}$, and a state information w^*. There exists $I \in \{1, ..., n\}$ such that A_1 has not asked the partial private key for ID_I, and the sign query on (m_I^*, ID_I^*, P_I^*). Let $I = 1$. Then the forged aggregate signature satisfies

$e(T^*, P) = e(P_{Pub}, \sum_{i=1}^{n} h_i^* Q_i^*) e(F^*, \sum_{i=1}^{n} R_i^* + g_i^* P_i^*)$ (1) where $Q_i^* = H_1(ID_i^*)$, $h_i^* = H_2(m_i^*, ID_i^*, P_i^*, R_i^*)$,

$g_i^* = H_3(m_i^*, ID_i^*, P_i^*, R_i^*)$, $F^* = H_4(w^*)$.

N_1 finds $(ID_i^*, \varepsilon_i^*, Q_i^*, c_i^*)$ from H_1^{list}, $(m_i^*, ID_i^*, P_i^*, R_i^*, \delta_i^*)$ from H_2^{list}, $(m_i^*, ID_i^*, P_i^*, R_i^*, \phi_i^*)$ from H_3^{list}, (w_i^*, μ_i^*, F_i^*) from H_4^{list} for all $i, 1 \le i \le n$. N_1 proceeds only if $c_1^* = 0$, $c_i^* = 1$ for all $i, 2 \le i \le n$. Otherwise, N_1 aborts.

From Eq. (1), it holds $e(T^*, P) = e(P_{Pub}, h_1^* Q_1^*) e(P_{Pub}, \sum_{i=2}^{n} h_i^* Q_i^*) e(F^*, \sum_{i=1}^{n} R_i^* + g_i^* P_i^*)$

$$\Rightarrow e(P_{Pub}, h_1^* Q_1^*) = e(T^*, P) \left(e(P_{Pub}, \sum_{i=2}^{n} h_i^* Q_i^*) e(F^*, \sum_{i=1}^{n} R_i^* + g_i^* P_i^*) \right)^{-1} \quad (2)$$

But $Q_1^* = \varepsilon_1^*(bP)$, $h_1^* = \delta_1^*$, $g_1^* = \phi_1^*$, $F^* = \mu^* P$, and for all $i, 2 \le i \le n$, $Q_i^* = \varepsilon_i^* P$, $h_i^* = \delta_i^*$, $g_i^* = \phi_i^*$.

Substituting in Eq. (2), we get $e(P_{Pub}, \delta_1^* \varepsilon_1^*(bP)) = e(T^*, P) \left(e(P_{Pub}, \sum_{i=2}^{n} \delta_i^* \varepsilon_i^* P) e(\mu^* P, \sum_{i=1}^{n} R_i^* + \phi_i^* P_i^*) \right)^{-1}$.

$$\Rightarrow e(aP, \delta_1^* \varepsilon_1^*(bP)) = e(T^*, P) \left(e(P, \sum_{i=2}^{n} \delta_i^* \varepsilon_i^* P_{Pub}) e(P, \left(\sum_{i=1}^{n} R_i^* + \phi_i^* P_i^* \right) \mu^*) \right)^{-1}.$$

$$\Rightarrow abP = (\varepsilon_1^* \delta_1^*)^{-1} \left[T^* - \left(\sum_{i=2}^{n} \delta_i^* \varepsilon_i^* P_{Pub} + K^* \mu^* \right) \right], \text{ where } K^* = \left(\sum_{i=1}^{n} \left(R_i^* + \phi_i^* P_i^* \right) \right).$$

Hence, the challenger N_1 can calculate $abP = (\varepsilon_1^* \delta_1^*)^{-1} \left[T^* - \left(\sum_{i=2}^{n} \delta_i^* \varepsilon_i^* P_{Pub} + \sum_{i=1}^{n} \left(R_i^* + \phi_i^* P_i^* \right) \mu^* \right) \right]$.

Theorem 2: The proposed certificateless aggregate scheme is existentially unforgeable against Type II adversary under the assumption that the CDH problem is intractable.

Proof: Due to the space constraint, we briefly prove the proposed scheme is existentially unforgeable against Type II adversary, and show how a CDH attacker N_2 uses Type II adversary A_2 to compute abP from (P, aP, bP).

Setup: Firstly, N_2 picks a random $\eta \in Z_q^*$ as the master-key, and sets system public key $P_{Pub} = \eta p$ and system parameters params $= (G_1, G_2, e, P, P_{Pub}, H_1, H_2, H_3, H_4)$. Then he sends params and the master key η to A_2. Since A_2 has access to the master-key, there is no need to handle H_1 as random oracle.

H_2 queries: There is a list H_2^{list} to record H_2 queries. When A_1 queries $H_2(m_i, ID_i, P_i, R_i)$, the same answer will be given if the query can be found on H_2^{list}. If the query cannot be found on H_2^{list}, N_1 picks a random $\delta_i \in Z_q^*$, adds $(m_i, ID_i, P_i, R_i, \delta_i)$ to H_2^{list} and then returns δ_i.

H_3 queries: There is a list H_3^{list} to record H_3 queries. When A_1 queries $H_3(m_i, ID_i, P_i, R_i)$, the same answer will be given if the query can be found on H_3^{list}. If the query cannot be found on H_3^{list}, N_1 picks a random $\phi_i \in Z_q^*$, adds $(m_i, ID_i, P_i, R_i, \phi_i)$ to H_3^{list} and then returns ϕ_i.

H_4 queries: There is a list H_4^{list} of tuple (w_i, μ_i, F_i) to record H_4 queries. When A_1 queries $H_1(w_i)$, the same answer will be given if the query can be found on H_4^{list}. Otherwise N_2 picks a random $\mu_i \in Z_q^*$, calculates $F_i = \mu_i(bP)$, adds (w_i, μ_i, F_i) to H_4^{list} and then returns F_i.

Public-Key queries: To answer a Public-Key query on ID_i, if the query can be found on K^{list}, the same answer will be given. Otherwise N_2 picks $x_i \in Z_q^*$ and flips a coin $c_i \in \{0,1\}$. If $c_i = 0$, N_2 returns $x_i(aP)$, adds $(ID_i, \perp, D_i, P_i, c_i)$ to list K^{list}. Otherwise, it calculates $P_i = x_i P$, and adds $(ID_i, x_i, D_i, P_i, c_i)$ to K^{list} and returns P_i.

Secret-Value queries: On receiving a Secret-Value query on ID_i, firstly, N_2 finds the tuple on K^{list}. If $c_i = 0$, N_2 aborts, otherwise, returns x_i.

Sign queries: When N_2 receives a Sign query on m_i by one user with identity ID_i, first N_2 recovers $(ID_i, x_i, D_i, P_i, c_i)$ from K^{list}, $(m_i, ID_i, P_i, R_i, \delta_i)$ from H_2^{list}, $(m_i, ID_i, P_i, R_i, \phi_i)$ from H_3^{list}, (w_i, μ_i, F_i) from H_4^{list}, then does as follows.

1. If $c_i = 0$, select $r_i \in Z_q^*$, set $R_i = r_i P - \phi_i P_i, T_i = r_i F_i + \delta_i D_i$ and output $\sigma_i = (R_i, T_i)$. Here δ_i, ϕ_i are taken from H_2^{list}, H_3^{list} respectively.
2. If $c_i = 1$, N_2 executes the standard sign algorithm to generate and output $\sigma_i = (R_i, T_i)$.

Forgery: As in Theorem 1, A_2 outputs a forged aggregate signature $\sigma^* = \{R_1^*, R_2^*, ..., R_n^*, T^*\}$ and hence N_2 calculates abP as follows. $e(F^*, R_1^* + g_1^* P_1^*) = e(T^*, P)\left(e(P_{Pub}, \sum_{i=1}^{n} h_i^* Q_i^*) e(F^*, \sum_{i=2}^{n} R_i^* + g_i^* P_i^*) \right)^{-1}$.

$$\Rightarrow e(\mu^* bP, R_1^* + g_1^* x_1^* aP) = e(T^*, P)\left(e(\eta P, \sum_{i=1}^{n} h_i^* Q_i^*) e(\mu^* bP, \sum_{i=2}^{n} R_i^* + g_i^* x_i^* P) \right)^{-1}.$$

$$\Rightarrow e(\mu^* bP, g_1^* x_1^* aP) = e(T^*, P)\left(e(\eta P, \sum_{i=1}^{n} h_i^* Q_i^*) e(\mu^* bP, \sum_{i=2}^{n} R_i^* + g_i^* x_i^* P) e(\mu^* bP, R_1^*) \right)^{-1}.$$

$$\Rightarrow abP = \left[T^* - \left(\sum_{i=1}^{n} \eta \delta_i^* Q_i^* + \sum_{i=2}^{n} \left(r_i^* + \phi_i^* x_i^* \right) \mu^*(bP) + \mu^* r_1^*(bP) \right) \right] (\mu^* \phi_1^*)^{-1}. \quad (3)$$

Hence the challenger N_2 can output abP as described in Eq. (3).

V. Efficiency Analysis

In this section we present the performance analysis of our proposed scheme. We compare our scheme with the schemes [5, 17, 23], which were compared in Kang et al.'s scheme. We consider the experimental results [24-27] to achieve the comparable security with 1024-bit RSA key, where the bilinear pairing (Tate pairing) is defined over the super singular elliptic curve $E/F_p : y^2 = x^3 + x$ with embedding degree 2 and the 160-bit Solinas prime number $q = 2^{159} + 2^{17} + 1$ with 512-bit prime number p satisfying $p + 1 = 12qr$. The details of these operations and their conversions are presented in Table no 1.

Table no 1: Notations and descriptions of various cryptographic operations and their conversions

Notations	Description
T_{MM}	Modular multiplication operation in Z_q^*
T_{SM}	Elliptic curve point multiplication, (Scalar multiplication in G_1), $T_{SM} = 29T_{MM}$
T_{BP}	Bilinear pairing in G_2 , $T_{BP} = 87T_{MM}$
T_H	Simple hash function which is negligible
T_{MTPH}	Map to point hash function, $1T_{MTPH} = 1T_{SM} = 29T_{MM}$
T_{PA}	Elliptic curve point addition in G_1 , $T_{PA} = 0.12T_{MM}$

Table no 2: Comparison of Computation Cost

Scheme	Sign. Cost	Agg. Verif. Cost	Total Cost
[17]	$4T_{SM} + 3T_{PA}$ $= 116.36T_{MM}$	$3T_{BP} + 2nT_{SM} + 2nT_{PA}$ $= (87 + 58.24n)T_{MM}$	$(203.36 + 58.24n)$ T_{MM}
[23]	$4T_{SM} + 2T_{PA}$ $= 116.24T_{MM}$	$4T_{BP} + 2nT_{SM} + (3n-3)T_{PA}$ $= (115.64 + 61n)T_{MM}$	$(231.88 + 61n)$ T_{MM}
[5]	$4T_{SM} + 2T_{PA}$ $= 116.24T_{MM}$	$4T_{BP} + nT_{SM} + (3n-1)T_{PA}$ $= (115.88 + 29.36n)T_{MM}$	$(232.12 + 29.36n)$ T_{MM}
Ours	$3T_{SM} + 2T_{PA}$ $= 87.24T_{MM}$	$3T_{BP} + 2nT_{SM} + (3n-1)T_{PA}$ $= (86.88 + 58.36n)T_{MM}$	$(174.12 + 58.36n)$ T_{MM}

We now analyze our improved aggregate signature scheme and compare it with [5, 17, 23]. L. Cheng *et al.* scheme [17] requires four scalar multiplications, three point additions to produce a digital signature and three bilinear pairings, 2*n* scalar multiplications and 2*n* point additions for aggregate signature verification. Thus, L. Cheng *et al.* scheme [17] needs $4T_{SM} + 3T_{PA} = 116.36T_{MM}$ for signature generation and $3T_{BP} + 2nT_{SM} + 2nT_{PA} = (87 + 58.24n)T_{MM}$ for aggregate signature verification. Hence the total computational cost of L.Cheng *et al.* scheme [17] is $(203.36 + 58.24n)T_{MM}$.

Similarly, H. Chen *et al.* scheme [23] requires four scalar multiplications, two point additions to produce a digital signature and four bilinear pairings, 2*n* scalar multiplications and (3*n*-3) point additions for aggregate signature verification. Thus, H. Chen *et al.* scheme [23] needs $4T_{SM} + 2T_{PA} = 116.24T_{MM}$ for signature generation and $4T_{BP} + 2nT_{SM} + (3n-3)T_{PA} = (115.64 + 61n)T_{MM}$ for aggregate signature verification. Hence the total computational cost of H. Chen *et al.* scheme [23] is $(203.36 + 58.24n)T_{MM}$.

Kang et al. scheme [5] requires four scalar multiplications, two point additions to produce a digital signature and four bilinear pairings, *n* scalar multiplications and (3*n*-1) point additions for aggregate signature verification. Thus, Kang *et al.* scheme [5] needs $4T_{SM} + 2T_{PA} = 116.24T_{MM}$ for signature generation and $4T_{BP} + nT_{SM} + (3n-1)T_{PA} = (115.88 + 29.36n)T_{MM}$ for aggregate signature verification. Hence the total computational cost of Kang *et al.* scheme [5] is $(232.12 + 29.36n)T_{MM}$.

Our improved scheme requires three scalar multiplications, two point additions to produce a digital signature and three bilinear pairings, 2*n* scalar multiplications and (3*n*-1) point additions for aggregate signature verification. Thus the proposed scheme needs $3T_{SM} + 2T_{PA} = 87.24T_{MM}$ for signature generation and $3T_{BP} + 2nT_{SM} + (3n-1)T_{PA} = (86.88 + 58.36n)T_{MM}$ for aggregate signature verification. Hence the total computational cost of improved scheme is $(174.12 + 58.36n)T_{MM}$.

The detailed comparison of our scheme with the existing schemes [5, 17, 23] is presented in terms of computational complexity and their results are presented in Table no 2. From Table no 2, it is clear that the individual/total computational cost of our scheme is much less than the schemes presented in [5, 17, 23]. Thus, our scheme has most excellent performance when compared to all other schemes.

VI. Conclusions

In this paper, we presented some drawbacks of B. Kang et al. scheme and described about the possibilities of forging the signature by an adversary. Later we presented the cryptanalysis of B. Kang et al. scheme by demonstrating how a Type II adversary can forge a legal aggregate signature on any message without

any access to user's secret information. We proved that Kang et al. scheme is not secure against malicious KGC attack. Furthermore, we presented an improved version of B. Kang et al. CLAS scheme. The proposed CLAS scheme can resist against various attacks and more efficient than well known existing schemes. The proposed scheme is proven secure in Random Oracle Model under the assumption of Computational Diffie-Hellman Problem is hard. The concrete security proof assures that our scheme is secure against Type I and Type II adversary. We compared our scheme with well known existing schemes. Efficiency analysis shows that our scheme is more efficient than existing schemes in terms of communication and computational costs.

References

[1] W. Diffie, and M.E. Hellman, "New Directions in Cryptography", *IEEE Transactions in Information Theory*, Vol. 22, pp. 644-654, 1976.

[2] A. Shamir, "Identity-based Cryptosystems and Signature Schemes", In: *Proc. of Advances in Cryptology, Crypto-1984*, Santa Barbara, CA, USA, Vol.196, pp. 47-53, 1984.

[3] S. S. Al-Riyami, and K. G. Paterson, "Certificateless Public key Cryptography", In: *Proc. of Advances in Cryptology, Asiacrypt-2003*, Taipei, Taiwan, Vol. 2894, pp. 452-473, 2003.

[4] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps", In: *Proc. of Advances in Cryptology, Eurocrypt-2003*, Warsaw, Poland, pp.416-432, 2003.

[5] B. Kang, M. Wang, and D. Jing, "An Efficient Certificateless Aggregate Signature Scheme", *Wuhan University Journal of Natural Sciences*, Vol. 22, No. 2, pp. 165-170, 2017.

[6] R. Castro, and R. Dahab, "Efficient Certificateless Signatures Suitable for Aggregation", *Cryptology eprint archive*, Report 2007/454, 2007.

[7] L. Zhang, and F. Zhang, "A New Certificateless Aggregate Signature Scheme", *Computation and Communication*, Vol. 32, No. 6, pp. 1079-1085, 2009.

[8] Z. Gong, Y. Long, X. Hong, and K. Chen, "Practical Certificateless Aggregate Signatures from Bilinear Maps", *Journal of Information Science and Engineering*, Vol. 26, pp. 2093-2106, 2010.

[9] Y. C. Chen, G. Horng, C. L. Liu, Y. Yu and C. S. Chan, "Efficient Certificateless Aggregate Signature Scheme", *Journal Electronic Science and Technology*, Vol.10, No.3, pp. 209-214,2012.

[10] M. Zho, M. Zhang, C. Wang and B. Yang, "CCLAS: A Practical and Compact Certificateless Aggregate Signature with Share Extraction", *International Journal of Network Security*, Vol.16, No.3, pp.174-181, 2014.

[11] F. Zhang, L. Shen, and G. Wu, "Notes on Security of Certificateless Aggregate Signature Schemes", *Information Science*, Vol.287, pp. 32-37, 2014.

[12] B. Kang and D. Xu, "A Secure Certificateless Aggregate Signature Scheme", *International Journal of Security and its Applications*, Vol.10, No.3, pp. 55-68, 2016.

[13] J. Kar, "Certificateless Aggregate Short Signature Scheme", e-print, IACR, 2016. https://eprint.iacr.org/2016/305.pdf

[14] L. Zhang, B. Qin, Q. Wu, and F. Zhang, "Efficient many-to-one Authentication with Certificateless Aggregate Signatures", *Computer Networks*, Vol. 54, No. 14, 2010.

[15] H. Xiong, Z. Guan, Z. Chen, and F. Li, "An Efficient Certificateless Aggregate Signature with Constant Pairing Computations", *Information Science*, Vol. 219, pp. 225-235, 2013.

[16] L. Cheng, Q. Wen, Z. Jin, H. Zhang, and L. Zhou, "Cryptanalysis and Improvement of a Certificateless Aggregate Signature Scheme", *Information Science*, Vol.295, pp. 337-346, 2014.

[17] Y. C. Chen, R. Tso, M. Mambo, K. Huang, and G. Horng, "Certificateless Aggregate Signature with Efficient Verification", *Security and Communication Networks*, 10.1002/ sec. 1166, 2014.

[18] H. Liu, S. Wang, M. Liang, and Y. Chen, "New Construction of Efficient Certificateless Aggregate Signatures", *International Journal of Security and its Applications*, Vol.8, No.1, pp. 411-422, 2014.

[19] H. Tu, D. He, and B. Huang, "Reattack of a Certificateless Aggregate Signature Scheme with Constant Pairing Computations", *The Scientific World Journal*, article ID343715, 2014.

[20] J. Deng, C. Xu, H. Wu, and L. Dong, "A New Certificateless Signature with Enhanced Security and Aggregation Version", *Concurrency and Computation: Practice and Experience* 10.1022/cpe.3551,2015.

[21] H. Chen, S. M. Wei, C. J. Zhu, and Y. Yang, "Secure Certificateless Aggregate Signature Scheme", *Journal of Software*, Vol. 26, No. 5, pp. 1173−1180, 2015.

[22] S. J. Horng, S F. Tzeng, P.H.Huang, X.Wang, T. Li, and M. K. Khan, "An Efficient Certificateless Aggregate Signature with Conditional Privacy-Preserving for Vehicular Sensor Networks", *Information Sciences*, Vol. 317, pp. 48-66, 2015.

[23] A. K. Malhi, and B. Shalini, "An Efficient Certificateless Aggregate Signature Scheme for Vehicular Ad-Hoc Networks", *Discrete Mathematics and Theoretical Computer Science*, DMTCS, Vol. 17:1, 2015, 317–338, 2015.

[24] P. Barreto, H. Y. Kim, and B. Lynn, "Efficient Algorithms for Pairing Based Cryptosystems", In: *Proc. of Advances in Cryptology, Crypto-2002*, Santa Barbara, CA, USA, pp. 354-368, 2002.

[25] X. Cao, W. Kou, and X. Du, "A Pairing-free Identity Based Authenticated Key Agreement Protocol with Minimal Message Exchanges", *Information Sciences*, Vol. 180, No.15, pp. 2895-2903, 2010.

[26] S. H. Tan, S. H. Heng, and B. M. Goi, "Java Implementation for Pairing-based Cryptosystems", In: *Proc. of International Conf. On Computational Science and its Applications, ICCSA-2010, Seoul, Korea*, Vol. 6019, pp. 188-198, 2010.

[27] MIRACL Library. Available at http://certivox.org/display/EXT/MIRACL.

N. B. Gayathri. " Cryptanalysis and Improvement of Kang et al. Certificateless Aggregate Signature Scheme." IOSR Journal of Computer Engineering (IOSR-JCE) 20.6 (2018): 60-68.